Evolving Data Sets to Highlight the Performance Differences between Machine Learning Classifiers

Thomas Raway[1] J. David Schaffer[2] Kenneth J. Kurtz[3] Hiroki Sayama[1,2]
[1] Department of Systems Science and Industrial Engineering
[2] Department of Bioengineering
[3] Department of Psychology
Binghamton University, State University of New York
P.O. Box 6000, Binghamton, NY 13902-6000, USA
{traway1, dschaffe, kkurtz, sayama}@binghamton.edu

ABSTRACT
We present a preliminary study to evolve data sets that maximize performance differences between multiple machine learning classifiers. The aim is to provide useful information towards the decision of which machine learning classifier to use given a particular data set. While literature already exists on comparing multiple classifiers across multiple pre-existing data sets, our approach is novel and unique in that we evolved completely new data sets designed to highlight the performance differences between supervised learning classifiers. By investigating these evolved data sets, we hope to add to the knowledge base concerning which classifiers are appropriate for specific real world classification tasks.

Categories and Subject Descriptors
I.5.2 [**Pattern Recognition**]: Design Methodology – *classifier design and evaluation*; D.2.2 [**Software Engineering**]: Design Tools and Techniques – *evolutionary prototyping*.

General Terms
Algorithms, Performance.

Keywords
Evolutionary Computation, Machine Learning, Complexity Measures

1. INTRODUCTION
A difficult, yet common, decision in the machine learning field is the selection of a classifier given a particular task. This choice is important due to classifier performance variability across multiple data sets and the relatively unknown relationship between data sets and classifiers that work well together. This variability can be described through the operational differences of classifiers. For example, if *classifier1* outperforms *classifier2* on a particular data set, then one may say its operation was more suited for that task. However, there also must exist data sets where *classifier2* has the ideal operation and will outperform *classifier1* [5]. Therefore we cannot assume a classifier will be dominant across all possible data sets. An appropriate goal then, is to select the classifier that will work best for the data set one is currently interested in.

GECCO'12 Companion, July 7–11, 2012, Philadelphia, PA, USA.
ACM 978-1-4503-1178-6/12/07.

One approach towards selecting a classifier given a particular data set is to utilize data set complexity measurements. For a machine learning practitioner these measurements are valuable as they capture important attributes of data sets that affect classifier performance (e.g. discriminative power of data set features, separability of classes, measures of geometry, and sparsity) [3]. Furthermore, by observing these complexity measurements across a multitude of data sets, each with classifier performance values, patterns of classifier dominance start to emerge [4]. Our research, meanwhile, is focused on the aptness of these complexity measurements, i.e., are the current set of measurements sensitive to the differences between data sets that cause variations in classifier performance?

We aim to help the answering of these questions through the evolution of artificial data sets where different classifier selection results in drastic variation in performance. The designing stage is carried out through an evolutionary process whereby populations of data sets compete with one another for the ability to survive and reproduce with variation. The fitnesses of these data sets are determined by how well they polarize two different classifiers, meaning high fitness is rewarded when both *classifier1* has high performance and *classifier2* has low performance. This process results in data sets where classifier choice is extremely important and therefore an ideal collection of complexity measurements should be sensitive to these situations. By evolving classifier selective data sets we hope to provide pertinent information regarding evaluations, additions, and changes to data set complexity measurements.

2. METHODS
2.1 Program Setup
All programming was done in Python. This decision was made in order to utilize Orange [2], a software suite available for Python which specializes in machine learning techniques, such as classification trees, k-nearest neighbors, support vector machines (SVM), etc. In addition, after applying a classifier to a target data set there are multiple measures of performance (e.g. accuracy, squared error, run time, ROC area, etc) that Orange provides, describing the success of a classifier. These built in capabilities, along with n-fold cross validation make it easy to measure a classifiers ability to generalize on new never before seen data. In essence, Orange allowed us to focus on the evolutionary aspect of our program instead of spending time building our own versions of these classifiers.

2.2 Evolutionary Strategy Setup

The creation of data sets is controlled by an Evolutionary Computational (EC) process most similar to Evolutionary Strategies (ES). This method was chosen due to its simplicity compared against the other members of the EC family, as there are usually fewer parameters to tune [1]. Each chromosome, meaning data set, of the population was encoded with a series of input feature vectors and classification designation pairs, with the number of these pairs being controlled by the user, which determines the number of samples in the data set. Other user defined parameters include number of features per sample and number of classes. At this early stage class distributions were even.

The reproductive process of each chromosome follows a $\mu+\lambda$ format where a parent chromosome produces λ number of children at each generation. Offspring creation was completed via the addition of random noise to a parent's input feature vectors. This mutation value was generated from a normal distribution with mean of zero and an adapting global standard deviation value. This standard deviation was controlled based on the survival rate of the offspring. If more than 1/5 of the offspring survived then the standard deviation was increased and vice versa.

The fitness of a data set was calculated based on the difference in performances between two classifiers. High fitness was rewarded to data sets that resulted in high performance with *classifier1* and low performance with *classifier2*. To maximize fitness a data set needs to be an appropriate match with *classifier1* and an inappropriate match with *classifier2*. It is important to note that the fitness value of a data set must be a comparison between at least two classifiers. This makes sure that the evolutionary process is designing data sets where classifier choice has a large impact. Otherwise, attempting to find high or low fitness data sets would result in overly simple or impossible tasks respectively.

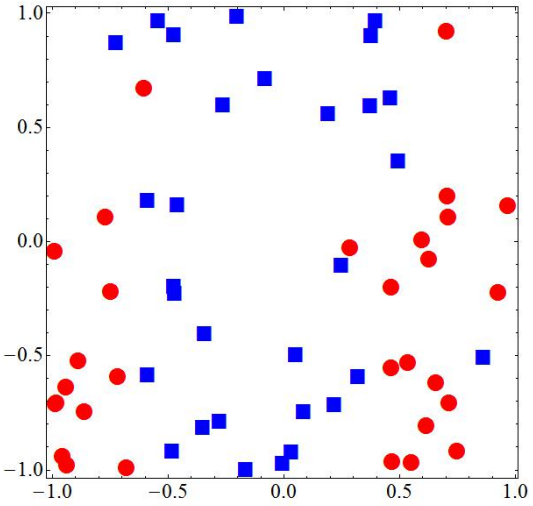

Figure 1. An evolved 2D data set maximizing the difference in performance between SVM-RBF and SVM-Linear. The axes show each sample's input feature values, while the shape/color of each sample determines the class.

We have conducted preliminary proof-of-concept experiments evolving two-dimensional data sets with pairs of classifiers. Figure 1 shows a data set evolved to maximize the difference in validation accuracy using 5-fold cross validation between two SVMs, the first with an RBF kernel and the second with a linear kernel. These two kernels were selected because linear kernels are unable to solve non-linearly separable problems, while the RBF kernel does not have this limitation. We anticipated the evolutionary process returning a data set that could not be solved linearly and Figure 1 shows that we achieved success in this regard.

3. DISCUSSION

As there is no universally best classifier across all possible data sets the choice of which classifier to use given a specific task is important. To aid in this decision data set complexity measurements are useful in quantifying attributes within a data set that are linked with classifier performance. But, how can we test the sensitivity of these measurements towards the differences between data sets that matter to classifiers? The evolutionary process we have outlined here facilitates the creation of artificial data sets that are the archetype of the importance for classifier choice. Furthermore, we believe these data sets would be an ideal testing ground for the evaluation, modification, and addition of data set complexity measurements.

Future modifications to this process include the shifting of control over certain parameters (e.g. data set size, number of features, class distribution) from the user to the evolutionary program. These are attributes of a data set that could matter to certain classifiers and therefore should be included within the data set search. Another possible change is in the reproduction stage, currently the data sets create offspring through pure mutation, however, recombination between chromosomes, allowing for whole sections of a data set to be copied, might be useful to the evolutionary process. And finally, changes to the number of classifiers within the fitness function could prove to be valuable. For example, comparing a classifier against an ensemble of classifiers, potentially resulting in data sets specifically aligned with the classifier of interest.

4. REFERENCES

[1] Beyer, Hans-Georg, and Schwefel, Hans-Paul. Evolution Strategies. *Natural Computing,* 1, (2002), 3-52.

[2] Curk, Tomaz et. al. Microarray Data Mining with Visual Programming. *Bioinformatics,* 21(3), 1 (Feb. 2005), 396-398.

[3] Ho, T.K. and Basu, Mitra. Complexity Measures of Supervised Classification Problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 24, (Mar, 2002).

[4] Mansilla, B., Ester and Ho, T.K. On Classifier Domains of Competence. *Proceedings of the 17th International Conference on Pattern Recognition.* (2004).

[5] Wolpet, H., David and Macready, G., William. No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation,* 1, (Apr 1997), 67-82.